



7

Linux Sharing Services

CERTIFICATION OBJECTIVES

- 7.01 Apache
- 7.02 Easy Squid Configuration
- 7.03 The Very Secure FTP Service

- 7.04 Mail Services
- ✓ Two-Minute Drill
- Q&A Self Test

Unix was developed by AT&T in the late 1960s and early 1970s, and was freely distributed among a number of major universities during those years. When AT&T started charging for Unix, a number of developers tried to create clones of this operating system. In one of these efforts, Linux was developed in the early 1990s

Many of these same universities were also developing the network that evolved into the Internet. With current refinements, this makes Linux perhaps the most Internet-friendly network operating system available. The extensive network services available with Linux are not only the tops in their field, they create one of the most powerful and useful Internet-ready platforms available today at any price.

Currently, Apache is the most popular Web server on the Internet. According to the Netcraft (www.netcraft.com) survey, which tracks the Web server associated with virtually every site on the Internet, Apache is currently used by more Internet Web sites than all the other Web servers combined. Naturally, Apache is included with RHEL 3.

RHEL 3 also includes a number of other services which I discuss in this chapter. These services include the Squid Proxy Server, which caches frequently used pages

INSIDE THE EXAM

Clients and Servers

On both exams, you'll need to configure a workstation on a network; this includes connecting to all types of network services. On the RHCE exam, you'll configure the corresponding network servers. In this chapter, you'll configure Web (HTTP/HTTPS), Web proxy, FTP, and various mail services. You need to know how to do the following five things with each of these services:

1. Install the RPM packages or package groups required by the service.

2. Configure the service to start automatically when you reboot Linux.
3. Set up the service for basic operation on your network.
4. Configure security by hosts and users for the service.
5. Use log files, service specific tools, and more to troubleshoot service problems.

If you're studying for the RHCT exam, read though this chapter. Part of the chapter addresses the network clients such as **lftp**, Web server, and e-mail that you may configure during your exam.

locally. There is the Very Secure FTP (vsFTP) daemon, which provides both basic and secure FTP server services. With vsFTP, you can secure users, directories, subdirectories, and files with various levels of access control.

Other standard services in the Linux/Unix world are e-mail services using the sendmail SMTP server and the POP and IMAP e-mail client services. These are the de facto standards for e-mail on the Internet.

This chapter deals with the basic concepts surrounding the use of these services, and a basic level of configuration. In all cases, the assumption is that your network settings are correct and functioning properly. If you're having problems with your network configuration, read Chapter 3.

As for the RHCE exam, you may have to configure or troubleshoot any of the services discussed in this chapter. So as you read this chapter and look through the configuration files and exercises, be willing to experiment. And practice, practice, practice what you learn.

CERTIFICATION OBJECTIVE 7.01

Apache

Apache is by far the most popular Web server today. Based on the HTTP daemon (**httpd**), Apache provides simple and secure access to all types of content, based on the regular HTTP protocol, as well as its secure cousin, HTTPS.

Apache is based on the server code developed by the National Center for Supercomputing Applications (NCSA). It included so many “patches” it became known as “a patchy” server. The Apache Web server continues to advance the art of the Web and provides one of the most stable, secure, robust, and reliable Web servers available. This server is under constant development by the Apache Software Foundation (www.apache.org).

While there are numerous other Web servers available, Apache is the only one described in the current curricula for RH133 and RH300. One alternative Web server is Tux, which you can configure separately. You can also configure it to work with Apache to optimize service for certain types of Web sites.

Apache is a service; basic Apache clients are Web browsers. Therefore, only those concerned with the RHCE should read this section. This provides the briefest of overviews on Apache. For more information, read the documentation online at httpd.apache.org/docs-2.0/, or *Apache Server 2.0: The Complete Reference* by Ryan Bloom (Mcgraw-Hill/Osborne, 2002).

exam

Watch

If you're studying for the exam using Red Hat Linux 9, the version numbers of Apache RPM packages are slightly different. However, if you're careful, the results should be the same (as per the Red Hat Exam Prep guide, we're

configuring the service for basic operation). There are defaults in some of the Red Hat Linux 9 versions of the Apache configuration files that you'll have to change or comment out in order to follow the discussion in this section.

Apache 2.0

Red Hat Enterprise Linux includes the latest major release of Apache. The version included with RHEL 3 is `httpd-2.0.46-25.ent`. While there are major differences with Apache version 1.3.x if you're a Web administrator or developer, the differences with respect to the RHCE exam are fairly straightforward. It supports the configuration for regular (HTTP) and secure (HTTPS) web services. Both are specified as requirements in the Red Hat Exam Prep guide.

- **New packages** If you're installing Apache from the Red Hat Installation RPMs, all the package names have changed. As you'll see in the following section, most start with `httpd`. Strangely enough, the username associated with Apache services is now `apache`.
- **Modular directive files** Basic directives, such as those based on Perl, PHP, or the Secure Socket Layer, are now configured separately in the `/etc/httpd/conf.d` directory.
- **Revised variables** Some variables have changed in the `httpd.conf` configuration file. For example, Apache listens for computers that are looking for Web pages on port 80. You can now change that port with the **Listen** variable.
- **Virtual hosts** Apache configuration is now normally based on virtual hosts, which allows you to host multiple Web sites on the same Apache server, using a single IP address.

You may see some of these characteristics if you use Apache 1.3.x; some of these features have been “backported” from Apache version 2.0.x.

Installation

The RPM packages required by Apache are included in the Web Server package group. As you'd have to install a total of 33 RPM packages, it's probably more efficient to install these packages using the Red Hat Package Management utility. If you selected the Web Server package group with its defaults when you installed RHEL 3, you can continue to the next section.

e x a m **watch**

If you know which of the 33 RPM packages you want to install, it may be slightly faster to install the Apache packages with the rpm command. However, if you use the Package Management utility, you can do other things while Apache is being installed.

Unless you have the Installation CDs available, you should start the Package Management utility, pointing to the networked source for the RHEL 3 installation files. If it's mounted on the /mnt/inst directory, you'd use the following command from a Linux GUI command line interface:

```
# redhat-config-packages --tree=/mnt/inst
```

For more information on this command, see Chapter 4.

Once the Package Management utility is open, select the Web Server package group. It's easiest to accept the defaults. As shown in Figure 7-1, when you select this package group, you'll be installing 27 RPM packages. In Red Hat Linux 9, there are fewer RPMs in the Web Server package group; the defaults still suffice.

Starting on Reboot

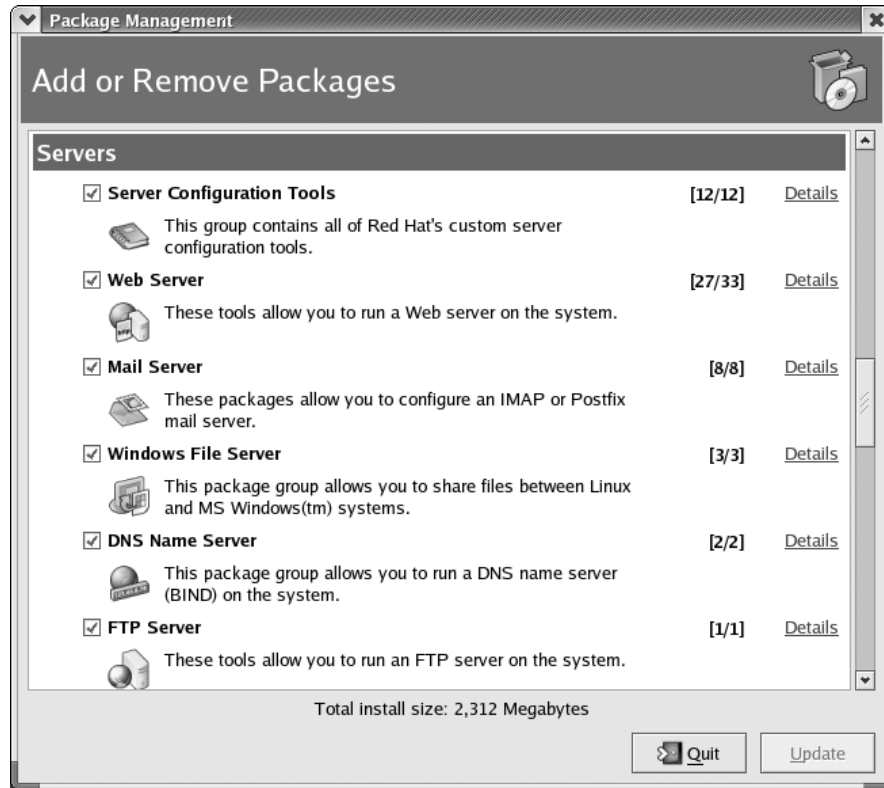
Once Apache is installed, you'll want to make sure it starts the next time you boot Linux. If it doesn't start when the person who grades your Red Hat exam reboots your computer, you may not get credit for your work on the Apache service.

The most straightforward way to make sure Apache starts the next time you boot Linux is with the **chkconfig** command. You'll need to set it to start in at least runlevels 3 and 5, with a command such as:

```
# chkconfig --level 35 httpd on
```

FIGURE 7-1

Installing from
the Package
Management
utility



To see if the **chkconfig** command worked, use the **--list** switch:

```
# chkconfig --list httpd
```

Normally to start services, it's best to use the associated script in the `/etc/rc.d/init.d` directory. There is an **httpd** script in that directory. However, Apache often starts and stops more gracefully with the following commands:

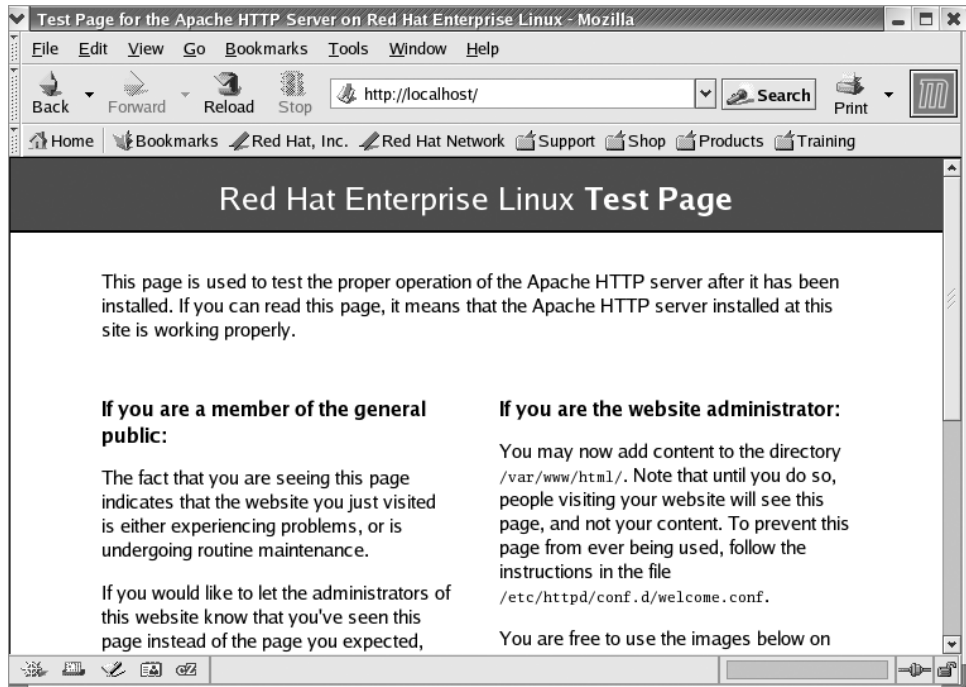
```
# apachectl stop
# apachectl start
```

Once you've got Apache running, start a Web browser and enter a URL of `http://localhost`. If Apache installation is successful, you should see the screen in Figure 7-2.

When you read the screen, you will see that RHEL 3 looks for Web page files in the `/var/www/html` directory. You can verify this with the `DocumentRoot` variable

FIGURE 7-2

The default
Apache Web
page



in the main Apache configuration file. If you want to create a custom error page, you can set it in the `/etc/httpd/conf.d/welcome.conf` file.

EXERCISE 7-1

Installing the Apache Server

In this exercise, you'll be installing all of the packages generally associated with the Apache server. Then you'll test and activate the result so that the Apache daemon, `httpd`, is active the next time you reboot Linux.

1. Open the GUI in your computer. If you aren't already in a graphical login manager, you can do so with the `startx` command.
2. Open a command line interface in the GUI. In the default GNOME desktop, right-click on the desktop and click New Terminal in the pop-up menu that appears. (In KDE, the Open Terminal command works as well.)

3. Mount to the source of RHEL 3 installation files. Refer to Chapter 2 for more information. For the purpose of this exercise, we'll assume that you're mounting on the local `/mnt/entr` directory.
 4. Start the Red Hat Package Management utility. Make sure it reads from the RHEL 3 installation files with the following command:

```
# redhat-config-packages --tree=/mnt/entr
```
 5. Locate the Web Server package group. Select it if required.
 6. Click Update. When you see the Completed System Preparation window, you may notice that there are more packages on the list than you selected in the Web server package group. Click Continue.
 7. When you see the Update Complete window, click OK. Click Quit to exit the Package Management utility.
 8. Back in the command line window, run the following command to see if Apache is already configured to start in any runlevels:

```
# chkconfig --list httpd
```
 9. Now, use the following command to make sure Apache starts in runlevels 3 and 5 the next time you boot Linux:

```
# chkconfig --level 35 httpd on
```
 10. Start the Apache service with the following command:

```
# /etc/rc.d/init.d/httpd start
```
 11. Start a Web browser such as Mozilla or Konqueror. Navigate to `http://localhost`. You should see the default Apache Web page for Red Hat Linux.
 12. Close your Web browser.
-

The Apache Configuration Files

There are two key configuration files for the Apache Web server: `httpd.conf` in the `/etc/httpd/conf` directory and `ssl.conf` in the `/etc/httpd/conf.d` directory. The default version of these files create a generic Web server service you can further customize and optimize, as desired. There are other configuration files, as shown in Figure 7-3, in two directories: `/etc/httpd/conf` and `/etc/httpd/conf.d`.

FIGURE 7-3

Apache
configuration files

```
[root@Enterprise3 root]# \ls -l /etc/httpd/conf
total 72
-rw-r--r--  1 root    root      35168 Sep 25 09:31 httpd.conf
-rw-r--r--  1 root    root     12959 Sep 25 09:31 magic
lrwxr-xr-x  1 root    root         37 Oct 23 16:12 Makefile -> ../../../../usr
/share/ssl/certs/Makefile
drwx-----  2 root    root      4096 Oct 23 16:12 ssl.crl
drwx-----  2 root    root      4096 Oct 23 16:12 ssl.crt
drwx-----  2 root    root      4096 Sep 25 09:31 ssl.csr
drwx-----  2 root    root      4096 Oct 23 16:12 ssl.key
drwx-----  2 root    root      4096 Sep 25 09:31 ssl.prm
[root@Enterprise3 root]# \ls -l /etc/httpd/conf.d
total 44
-rw-r--r--  1 root    root      517 Sep  8 09:17 authz_ldap.conf
-rw-r--r--  1 root    root     1800 Sep  8 10:07 perl.conf
-rw-r--r--  1 root    root      459 Sep  5 13:04 php.conf
-rw-r--r--  1 root    root     1276 Jul  3 06:14 python.conf
-rw-r--r--  1 root    root      180 Sep 25 09:31 README
-rw-r--r--  1 root    root       96 Mar 24 2003 squirrelmail.conf
-rw-r--r--  1 root    root     11316 Sep 25 09:31 ssl.conf
-rw-r--r--  1 root    root       333 Aug  2 06:17 webalizer.conf
-rw-r--r--  1 root    root       299 Sep 25 09:31 welcome.conf
[root@Enterprise3 root]#
```

on the


Previous versions of Apache, 1.3.x and below, required two other Apache configuration files in the same directory, *access.conf* and *srn.conf*. Even though these files were essentially blank in later versions of Apache 1.3.x, they were still required. These files are no longer required in any way in Apache 2.x.

The generic configuration file is *httpd.conf*. If you're required to configure a Secure Web server on the RHCE exam, you'll also need to configure the *ssl.conf* configuration file.

Basic Apache Configuration for a Simple Web Server

As described earlier, Apache looks for Web pages in the directory specified by the *DocumentRoot* variable. In the default *httpd.conf* file, this variable points to the */var/www/html* directory.

In other words, all you need get your Web server up and running is to transfer Web pages to the */var/www/html* directory. The default *DirectoryIndex* variable looks for an *index.html* Web page file in this directory. You can test this for yourself by copying the default Mozilla home page file, *index.html*, from the */usr/share/doc/HTML* directory.

If you have an **iptables** firewall on your computer, you'll need to disable it at least for TCP/IP port 80. Chapter 10 describes this process in detail.

The base location of configuration and log files are determined by the `ServerRoot` variable. The default value from `httpd.conf` is:

```
ServerRoot "/etc/httpd"
```

You'll note that the main configuration files are stored in the `conf` and `conf.d` subdirectories of the `ServerRoot`. If you run the `ls -l /etc/httpd` command, you'll find that Red Hat links `/etc/httpd/logs` to the directory with the actual log files, `/var/log/httpd`.

Analyzing the Default Apache Configuration

Apache comes with a well-commented set of default configuration files. In this section, we'll look at the key commands in the `httpd.conf` configuration file, in the `/etc/httpd/conf` directory. Browse through this file in your favorite text editor or with a command such as **less**. Before we begin our analysis, keep two things in mind:

- If you configure Apache with the Red Hat HTTP tool (**redhat-config-httpd**), it overwrites any changes that you may have made with a text editor.
- The main Apache configuration file incorporates the files in the `/etc/httpd/conf.d` directory with the following directive:

```
Include conf.d/*.conf
```

There are a couple of basic constructs in `httpd.conf`. First, directories, files, and modules are configured in containers. The beginning of the container starts with the name of the directory, file, or module to be configured, contained in directional brackets (`< >`). Examples of this include:

```
<Directory "/var/www/icons">
<Files ~ "^\.ht">
<IfModule mod_mime_magic.c>
```

The end of the container starts with a forward slash. For the above examples, the end of the container would be:

```
</Directory>
</Files>
</IfModule>
```

Next, Apache includes a substantial number of directives, which are commands that Apache can understand that have some resemblance to English. For example, the `ExecCGI` directive allows executable CGI scripts

Apache Security

You can modify the `httpd.conf` configuration file to secure the entire server or manage security on a directory-by-directory basis. Directory controls secure access by the server, as well as users who connect to the Web sites on the server. To explore the basics of Apache security, start with the first default active line in `httpd.conf`:

```
ServerTokens OS
```

This line looks deceptively simple; it limits what readers see about your Web server when you browse to a nonexistent page. If you don't use this command, outsiders can see if you've loaded modules such as Perl, Python, and PHP. Sharing this knowledge can make your system more vulnerable. You can restrict access to the root directory on your Web server, as shown below:

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
```

This configures a very restrictive set of permissions. The **Options FollowSymLinks** line supports the use of symbolic links for Web pages. The **AllowOverride None** line disables any `.htaccess` files. Otherwise, `.htaccess` can allow others to administer your server, starting from your computer's root directory.

You can improve this further by limiting access to all but desired users, such as those within your company, by adding the following commands to the `<Directory />` container:

```
Order deny,allow
Deny from all
```

The next excerpt limits access to `/var/www/html`, which corresponds to the default `DocumentRoot` variable:

```
<Directory /var/www/html>
    Options Indexes FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

You'll note that the **Options** directive has changed; the **Indexes** setting allows readers to see a list of files on your Web server if there is no `index.html` file in the `DocumentRoot`

directory. The **Order** and **Allow** lines shown here allow all users to access the Web pages on this server.

Finally, the **Listen** directive defines the IP address and TCP/IP port for this server. For example, the default shown here means that this server will work with every computer that requests a Web page from any of the IP addresses for your computer on the standard TCP/IP port, 80.

```
Listen 0.0.0.0:80
```

If you have more than one IP address on your computer, you can use this directive to limit this Web server to one specific IP address. For example, if you've set up an intranet on this Web server, you could use the IP address that connects to your private network here.

exam

Watch

The Red Hat Exam Prep guide suggests that you need to be ready to configure a regular HTTP and a secure HTTPS Web server.

If you're also setting up secure Web services, there's a second **Listen** directive in the `ssl.conf` file in the `/etc/httpd/conf.d` directory. The data from this file is automatically incorporated into your Apache configuration. It includes the following directive, which points to the default secure HTTP (HTTPS) port for TCP/IP, 443:

```
Listen 0.0.0.0:443
```

Virtual Hosts

Another useful feature of Apache 2.0 is its ability to manage Web sites using a single IP address. You can do so by creating multiple virtual hosts on the same Web server. You can configure virtual hosts for regular Web sites in the main Apache configuration file `/etc/httpd/conf/httpd.conf`. In that way, you can link multiple domain names such as `www.example.com` and `www.mommabears.com` to the same IP address on the same Apache server.



The `www.example.com` domain name cannot be registered and is officially reserved by the Internet society for documentation.

You can also create multiple secure Web sites which conform to the HTTPS protocol, by configuring virtual hosts in the `/etc/httpd/conf.d/ssl.conf` configuration file. While the details vary, the basic directives that you'd use in this file are the same.

Section 3 of the default `httpd.conf` includes sample commands which you might use to create one or more virtual hosts. To activate the Virtual Host feature, you'll

want to activate this line. If you're using a name-based host, leave the asterisk after this directive. Otherwise, set the IP address for your interface.

```
#NameVirtualHost *:80
```

exam

Watch

If you're required to create a virtual host for a secure Web site, you'll need a second NameVirtualHost directive for the HTTPS port, 443.

It's often more reliable to substitute the IP address, as it avoids the delays sometimes associated with name resolution through a DNS server.

You should already know that TCP/IP port 80 is the default for serving Web pages. If you want to direct all requests to IP address 192.168.30.2 on port 80, you can substitute `<VirtualHost 192.168.30.2:80>` it for the first line shown below:

```
#<VirtualHost *>
#     ServerAdmin webmaster@dummy-host.example.com
#     DocumentRoot /www/docs/dummy-host.example.com
#     ServerName dummy-host.example.com
#     ErrorLog logs/dummy-host.example.com-error_log
#     CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>
```

Don't forget to uncomment the commands shown by removing the # in front of each line. As you can see, it includes a number of directives from the main part of the configuration file. Here are the highlights of this container:

- You'll want to substitute the Web server IP address and TCP/IP port for the asterisk (*) in the `<Virtual Host *>` command. The standard HTTP port is 80; the standard HTTPS port is 443.
- Error messages are sent to the e-mail address defined by `ServerAdmin`.
- The Web pages can be stored in the `DocumentRoot` directory.
- Clients can call this Web site through the `ServerName`.
- The `ErrorLog` and `CustomLog` files use the *relative* log directory—relative to the `ServerRoot`. In other words, you can find these files in the `/etc/httpd/logs` directory. As noted earlier, this directory is linked to `/var/logs/httpd`.

Substitute the IP addresses, domain names, directories, and e-mail addresses of your choice. Create the `DocumentRoot` directory if required. You can test the syntax of what you've done with the following command:

```
# httpd -t
```

and Apache will verify your configuration or identify specific problems. When you run this command on the default configuration, you'll get the following message:

```
httpd: Could not determine the server's fully
qualified domain name, using 127.0.0.1 for ServerName
Syntax OK
```

This message is nothing to worry about during initial testing; it just notes that you haven't set a fully qualified domain name for the overall Web server through the **ServerName** directive.

Secure Virtual Hosts

If you're configuring a secure Web server, which conforms to the HTTPS protocol, Red Hat provides a different configuration file for this purpose: `ssl.conf` in the `/etc/httpd/conf.d` directory. If you don't see this file, you need to install the `mod_ssl` RPM. Before you begin editing this file, make sure the following **Listen** directive is active:

```
Listen 0.0.0.0:443
```

Later in this file, pay attention to the `<VirtualHost _default_:443>` container. You'll need to make sure that all of the directives between this line and the `</VirtualHost>` line at end of the file are included in some way. If you accidentally leave out some of these commands, you'll end up with a non-working Web server. You can replace `_default_` in the `VirtualHost` container with the IP address of your local secure server.

Follow the same guidelines as described for the regular virtual host. Ideally, you should configure a `DocumentRoot` in a different directory. You'll also need to add an `index.html` file to this directory. One possible `DocumentRoot` directive is:

```
DocumentRoot /www/secure/dummy-host.example.com
```

You'll also need to add a `ServerName` directive, pointing to the secure HTTP port, 443. For the previously noted Virtual Host domain, that would lead to the following directive:

```
ServerName dummy-host.example.com:443
```

A Basic Web Page

You may need to create some `index.html` files. You could use Apache's default Web page. You can change this or any other Web page with a text- or HTML-specific editor. If you need to create a simple Web page during the exam, you don't have to use HTML tags. The Red Hat exams aren't intended to test your knowledge of HTML code.

Alternatively, you may be able to create a basic `index.html` page on the RHCE exam, as a simple text file. For example, you might save the following line in the text editor of your choice as `index.html`. You could then copy it to the appropriate `DocumentRoot` directory.

```
This is a simple Web page
```

Checking Syntax

You can also check the work that you've done to create virtual hosts with the following command:

```
# httpd -S
```

Assuming there were no problems, you should be able to start your Web server and connect to your local service with a browser request. (This command does not work in Red Hat Linux 9.)

e x a m

W a t c h *Be prepared to create multiple Web sites on an Apache Web server using virtual hosts. If you're required to do so on your exam, create a separate VirtualHost container for this purpose.*

The beauty of `VirtualHost` containers is that you can repeat the process. With name-based virtual hosting, you can set up as many Web sites on your Apache server as your computer can handle. All you require is one IP address. When you set up your new `VirtualHost` container, make sure to revise at least the `ServerName`, the locations of the log files, and the `DocumentRoot`.

Host-Based Security

You can add the **Order**, **allow**, and **deny** directives to regulate access based on hostnames or IP addresses. This basic command reads the **deny** directive first:

```
Order deny,allow
```

You can **deny** or **allow** from various forms of hostnames or IP addresses. For example, the following directive denies access from all computers in the `osborne.com` domain:

```
Deny from osborne.com
```

It's preferable to use IP addresses, so communication with a DNS server does not slow down your Web server. The following example directives use a single IP address;

alternatively, you can set up the 192.168.30.0 subnet in partial, netmask, and CIDR (Classless InterDomain Routing) notation:

```
Deny from 192.168.30.66
Allow from 192.168.30
Deny from 192.168.30.0/255.255.255.0
Allow from 192.168.30.0/24
```

User-Based Security

You can limit access to your Web sites to authorized users with passwords. As I'll describe shortly, these passwords can be different from the regular password database on your Linux computer.

If you want to configure User-based security for the virtual Web site described earlier, you'll need to set up a <Directory> container for the directory specified by the DocumentRoot. In this case, it's /www/docs/dummy-host.example.com. You'll want several commands in the <Directory> container:

- To set up basic authentication, you'll need an **AuthType Basic** command.
- To describe the site to requesting users, you can include an **AuthName "some comment"** command.
- To refer to a Web server password database named /etc/httpd/webpass, you'll need a **AuthUserFile /etc/httpd/webpass** command.
- To limit the site to a single user named engineer1, you could add a **Require user engineer1** command.
- Alternatively, to limit the site to a group as defined in /etc/httpd/webgroup, you'd add the **AuthGroupFile /etc/httpd/webgroups** command. You'll also need a **Require group Design** command, where Design is the name of the group specified in webgroups.

Here's an example of code that I've added after the <Virtual Host> container:

```
<Directory "/www/docs/dummy-host.example.com">
    AuthType Basic
    AuthName "Internal Use Only"
    AuthUserFile /etc/httpd/webpass
    Require user michael
</Directory>
```

When properly configured, the next time you try accessing the dummy-host.example.com Web site in the Mozilla browser, you're prompted for a username and password as shown in Figure 7-4.

FIGURE 7-4

A secured
Web site



Configuring Web Passwords

To configure passwords for a Web site on your server, you need to create a separate database of usernames and passwords. As the **useradd** and **passwd** command is used for regular users, the **htpasswd** command is used to set up usernames and passwords for your Web server.

For example, if you want to create a database file named `webpass` in the `/etc/httpd` directory, you'd start with the following command:

```
# htpasswd -c /etc/httpd/webpass engineer1
```

The **-c** switch creates the specified file, and the first user is `engineer1`. You're prompted to enter a password for `engineer1`. Users in the `webpass` database do not have to have a regular account on your Linux computer.

Once you've created the database file, you can add to it without the **-c** switch; for example, the following command prompts you for a password for `drafter1` before adding it to your `webpass` database:

```
# htpasswd /etc/httpd/webpass drafter1
```

To set up access for more than one user, you'll also need a group file. For the example described in the previous section, you can set up a `webgroup` file in the same directory with the following line of users:

```
Design: engineer1 drafter1 lead1
```

Web Access to Home Directories

One useful option through Apache is access to user's home directories. If you change the following directive from **UserDir disable** to:

```
UserDir public_html
```

anyone will have access to Web pages that a user puts in his or her ~/public_html directory. For example, if you have a user named michael, he can create a /home/michael/public_html directory, and add the Web pages of his choice.

However, this requires a bit of a security compromise; you need to make michael's home directory executable for all users. This is also known as 701 permissions, which you can configure with the following command:

```
# chmod 701 /home/michael
```

Control Through .htaccess

There is a way to override inherited permissions in any subdirectory. If you're willing to set the following command in the <Directory> containers of your choice:

```
AllowOverride Options
```

you can configure .htaccess files to override previously set permissions. In this way, you can customize permissions on different virtual hosts. For example, you can configure an intranet site limited to your employees and an Internet site for general access on the same Apache server.

You can store an .htaccess file in every Web directory. While you can configure all types of Apache directives in this file, the Red Hat Exam Prep guide suggests that all you need to do is configure host-based and user-based security for the service.

Options and .htaccess

There are many ways to activate the **Options** directive. The simplest two are **Options None** and **Options All**. They would do the following:

None	For no custom options in force
All	Allows all options except MultiViews

You can also set finer options with keywords. Some examples include:

ExecCGI	Permits Web pages to run CGI scripts
FollowSymLinks	Permits symbolic links to directories outside of DocumentRoot
Includes	Allows server-side includes
Indexes	Permits FTP-style directory indexing

The **Indexes** directive controls directory indexing, which is the name for file lists that are generated automatically by Apache.

Executable Files in Apache

You have many different ways to set up Apache resources. Some might be available in different languages, different media types, or more. When you set up multiple resources, Apache can choose depending on the browser-supplied preferences for media type, languages, character set, and encoding.

You can use the **ScriptAlias** variable for directories with executable CGI files. An **Alias** variable essentially links one file or directory to another. The **ScriptAlias** variable links the default cgi-bin directory to `/var/www/cgi-bin`. You can set up CGI scripts in a different directory and change the reference accordingly.

```
ScriptAlias /cgi-bin/ "/var/www/cgi-bin"
```

```
<Directory /var/www/cgi-bin>
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
```

This excerpt from the default `/etc/httpd/conf/httpd.conf` file first identifies the directory with server scripts. The user `apache` is allowed to run these scripts. Permissions for others are adjusted through the `<Directory /var/www/cgi-bin>` container. The **AllowOverride None** command prevents regular users from changing permissions/settings in that directory. Otherwise, smarter users could read the CGI files in your directory, potentially compromising the security of your Web server. The **Options None** line prevents other users from running CGI scripts in the given directory. The **Order allow,deny** command sets up authorization checks; **Allow from all** lets all users run scripts in this directory.

The **Alias /manual "/var/www/manual"** command allows clients to read documentation stored in the `/var/www/manual` directory. If the name of your Web site is `www.example.com`, the alias allows clients to read the documentation by navigating to `www.example.com/manual`.

The **Alias /icons "/var/www/icons"** command identifies a directory for icons on your Web site. If the name of your Web site is `www.example.com`, you can set up the icons in HTML code on your page in the `www.example.com/icons` directory. You can then store the icons on your computer in the `/var/www/icons` directory.

Finally, you can add access control for any other directories available via your Web interface. Just wrap the directory you wish to control in a `<Directory /path/to/dir> . . . </Directory>` container and set the access restrictions you need. The command shown below applies the **Deny** directive first:

```
<Directory /path/to/your/directory/goes/here/>
    Options Indexes FollowSymLinks
    Order deny,allow
    Deny from .evil.crackers.net
    Allow from .yourdomain.net
</Directory>
```

Apache Log Files

As described earlier, the log files in `httpd.conf` are configured in the `/etc/httpd/logs` directory. It's linked to `/var/logs/httpd`. Access to your Web server is logged in the `access_log` file; errors are recorded in the `error_log` file. If you want more detail about your Web site for tuning or statistical reasons, you can have the Web server generate more information, generate separate log files for each virtual Web site, and create new log files at different frequencies (such as daily, weekly, or monthly).

There are standard log file formats. For more information, take a look at the **LogFormat** directive as shown in Figure 7-5. Four different formats are shown: combined, common, the referrer (the Web page with the link used to get to your site), and the agent (the user's Web browser). The first two **LogFormat** lines include a number of percent signs followed by lowercase letters. These variables determine what goes into the log.

You then use the **CustomLog** directive to select a location for the log file (for example, `logs/access_log`) and which log file format you wish to use (for example, common). For more information on log files and formats, refer to httpd.apache.org/docs-2.0/logs.html.



Some Web hit log analyzers have specific requirements for log file formats. For example, the popular Open Source tool *awstats* (advanced Web Stats) requires the combined log format. It will fail to run if you leave the default common format. *Awstats* is a great tool for graphically displaying site activity. You can download it from a site such as www.sourceforge.net.

FIGURE 7-5Customized
Apache logs

```
LogLevel warn

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
#
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

#
# The location and format of the access logfile (Common Logfile Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here. Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
#
# CustomLog logs/access_log common
CustomLog logs/access_log combined
```

Apache Troubleshooting

When you install the right Apache packages, the default configuration normally creates a running system. But if you're setting up a real Web site, you probably want more than just the test page. Before you start changing the configuration, back up the `httpd.conf` Apache configuration file. If something goes wrong, you can then always start over. Some Apache errors fall into the following categories:

- **Error message about an inability to bind to an address** Another network process may already be using the default http port (80). Alternatively, your computer is running httpd as a normal user (not the user `apache`) with a port below 1024.
- **Network addressing or routing errors** Double-check your network settings. For more information on configuring your computer for networking, see Chapter 4.
- **Apache isn't running** Check the error message when you use the `apachectl` command to start or restart the Apache server. Check the `error_log` in the `/var/log/httpd` directory.

- **Apache isn't running after a reboot** Run `chkconfig --list httpd`. Make sure Apache (httpd) is set to start at runlevels 3 and 5 during the boot process with the following command:

```
# chkconfig --level 35 httpd on
```

- **You need to stop Apache** Send the parent process a **TERM** signal, based on its PID. By default, this is located in `/var/run/httpd.pid`. You kill Apache with a command such as:

```
#kill -TERM `cat /var/run/httpd.pid`
```

Alternatively, you can use the **apachectl stop** command.

EXERCISE 7-2

Updating a Home Page

In this exercise, you'll update the home page associated with your Web site on the Apache server. You can use these techniques to copy the actual HTML formatted pages that you'll need for your Web site.

1. Start the Apache Web server with the default configuration.
2. Copy an HTML file such as `/var/www/error/noindex.html` to `/var/www/html/index.html`.
3. Edit the file `/var/www/html/index.html`.
4. Change the title of the page to reflect your personal or corporate name.
5. Use a Web browser such as Mozilla to connect to localhost (or 127.0.0.1).

on the
Job

Apache administration is a necessary skill for any Linux system administrator. You should develop the ability to install, configure, and troubleshoot Apache quickly. You should also be able to set up and customize virtual Web sites, which will make you a more effective Webmaster. You can test your skills using the exercise that follows.

EXERCISE 7-3

Set Up a Virtual Web Server

In this exercise, you'll set up a Web server, with a virtual Web site.. You can use this technique with different directories to set up additional Web sites on the same Apache server.

1. Back up your `httpd.conf` file.
 2. Add a virtual Web site for the fictional company SnoBard, called `www.snobard.net`.
 3. Create a DocumentRoot directory called `/snobard`. (Don't forget to create this directory on your system as well.)
 4. Open the `/snobard/index.html` file in your text editor. Add a simple line in text format such as:

```
This is the placeholder for the SnoBard Web site.
```
 5. Save this file.
 6. Open the browser of your choice. Test access the configured Web site (`www.snobard.com`) and the Web site on the localhost computer.
-

The Red Hat `httpd` Configuration Tool

Red Hat has recently introduced its own graphical configuration tool for Apache, **redhat-config-httpd**, which you can install from the RPM of the same name. Before using this tool, back up your current `/etc/httpd/conf/httpd.conf` configuration file. Any changes that you make with this tool overwrite this file.

You will find that **redhat-config-httpd** is a straightforward tool, with four different tabs that can help you configure the `httpd.conf` configuration file. You can also open this tool in the GUI with the Main Menu | System Settings | Server Settings | HTTP command. However, you cannot use it to edit the `ssl.conf` configuration file, and therefore should not use it to create a secure (HTTPS) Web server.

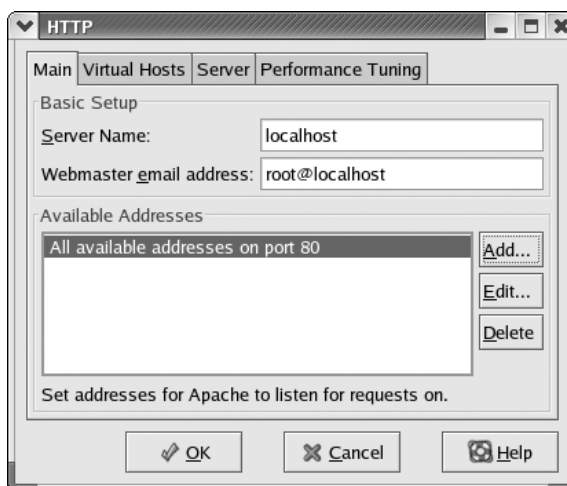
However, it may be useful to practice creating virtual hosts with this utility, so I've described the tabs shown in Figure 7-6 here:

- **Main** The Main tab allows you to set basic parameters for your Apache server, including the `ServerName`, the Webmaster e-mail address, and the `Listen` variable.
- **Virtual Hosts** The Virtual Hosts tab permits you to set the properties for different Web sites that you host on your Apache server. This includes the `DocumentRoot`, basic HTML filenames and locations, SSL support, basic log file configuration, CGI script variables, and default directories.
- **Server** The Server tab enables you to set the basic lock and PID files, as well as the user and group associated with the `httpd` service. In most cases, you should not have to change these settings.
- **Performance Tuning** The Performance Tuning tab allows you to set basic connection parameters.

Even if you do master this tool for configuring a regular Web server, you'll still need to manually edit the `ssl.conf` file to create a secure Web server. And I believe that it's easier to learn and faster to edit the Apache configuration file, `/etc/httpd/conf/httpd.conf`, directly in a text editor. Remember, time may be of the essence when you take the RHCE exam.

FIGURE 7-6

The Apache configuration tool, Main tab



CERTIFICATION OBJECTIVE 7.02

Easy Squid Configuration

Squid is a high-performance HTTP and FTP caching proxy server. It is also known as a Web proxy cache. It can make your network connections more efficient. As it stores data from frequently used Web pages and files, it can often give your users the data they need without having to look to the Internet.

Studies on very busy networks suggest that a Squid server can reduce the size, or bandwidth, of your Internet connection by 10–20 percent. That can lead to considerable savings for larger offices.

Squid conforms to the original Harvest Cache architecture and uses the Inter-Cache Protocol (ICP) for transfers between participating peer and parent/child cache servers. It can be used either as a traditional caching proxy or as a front-end accelerator for a traditional Web server. Squid accepts only HTTP requests, but speaks FTP on the server side when FTP objects are requested. You can join the worldwide hierarchy of Harvest Cache sites; see www.ircache.net for more information.

Required Packages for Squid

There is one Squid RPM package, `squid-2.5.STABLE3-3.3E`. It is installed by default when you install the Web Server package group. So if you've installed Apache and have not tinkered with the defaults, the Squid RPM should also be installed on your computer. This RPM package installs a substantial number of files and scripts; some of the key files include the following:

- `/etc/rc.d/init.d/squid` Start/stop script
- `/etc/squid/` Configuration directory
- `/usr/share/doc/squid-2.5.STABLE3` Documentation, mostly in HTML format
- `/usr/lib/squid/` Support files and internationalized error messages
- `/usr/sbin/squid` Main Squid daemon
- `/var/log/squid/` Log directory
- `/var/spool/squid/` Cache directory (Hundreds of MB and maybe more in many hashed directories)

There are minor differences between the versions of Squid on RHEL 3 and Red Hat Linux 9. The changes that you'll make are identical; however, the locations of each command that you'll change or add vary slightly.

Starting Squid on Reboot

The Squid Web Proxy is not started by default on RHEL 3. To do so, you'll want to activate it using a command such as **chkconfig** or the Service Configuration utility described in Chapter 4. The easiest way to set Squid to start the next time you boot Linux is with the following command:

```
# chkconfig --level 35 squid on
```

When the Squid Proxy Server starts for the first time, the `/etc/rc.d/init.d/squid start` script starts the Squid daemon. Squid runs as a caching proxy server on port 3128. You can then set up Web browsers on your LAN to point your computer to Squid on port 3128 instead of an external network such as the Internet. In that way, Squid would get the first chance at serving the needs of users on your network.

Configuration Options

You can configure and customize the way Squid operates through its configuration file, `/etc/squid/squid.conf`. The default version of this file includes a large number of comments that can help you tune and secure Squid. Since it has more than 3,000 lines, this isn't the easiest file to review. However, you only need to add three lines to this file before you activate Squid. For example, if the name of the local computer is Enterprise, you'd add the following line:

```
visible_hostname Enterprise
```

Next, to support regular Web (HTTP) access, you'll need to set the `http_access` variable to allow some arbitrary name:

```
http_access allow local_net
```

Next, you'll need to add your local network to the Squid Access Control List. This particular command line uses the `local_net` setting that you just created, and adds the IP addresses of a private network that I've used:

```
acl local_net src 192.168.30.0/255.255.255.0
```

Now you can save your changes and exit from the `squid.conf` configuration file. You can then create the basic cache directories in `/var/spool/squid` with the following command:

```
# squid -z
```

Finally, start the Squid service for the first time with the appropriate **service** command:

```
# service squid start
```

While you're unlikely to have a chance to configure more than one computer with Squid, its power is in connecting the cache from multiple servers. You can configure this with the `cache_peer` lines, which specify parent and sibling Squid cache servers. If your Linux computer is part of a group of Squid servers in a Harvest Cache, these lines allow your Squid servers to check these other Squid servers before going to the Internet.

Squid first checks its own cache and then queries its siblings and parents for the desired object such as a Web page. If neither the cache host nor its siblings have the object, it asks one of its parents to fetch it from the source. If no parent servers are available, it fetches the object itself.



Squid can greatly improve the performance of a corporate intranet. If your company has many employees who surf the Net, a Squid server can reduce your network connection costs by decreasing the bandwidth you need for your Internet connection.

EXERCISE 7-4

Configuring Squid to Act as a Proxy Server

This exercise assumes you have a LAN. One of the computers on the LAN is also a server that is connected to the Internet. It also assumes that you've already installed the Squid RPM. Then you can configure Squid to act as a proxy for Web and FTP service for your LAN.

1. Open the Squid configuration file, `/etc/squid/squid.conf` in a text editor.
2. Add the name of your computer to this file. Add the following command near the comments associated with **visible_hostname**:

```
visible_hostname computername
```

3. Add an **http_access** command to allow access from your local network. You can set an arbitrary name of your choice for the network; but you'll need to

use it in the command afterwards. Locate the command near the other **http_access** commands in this file.

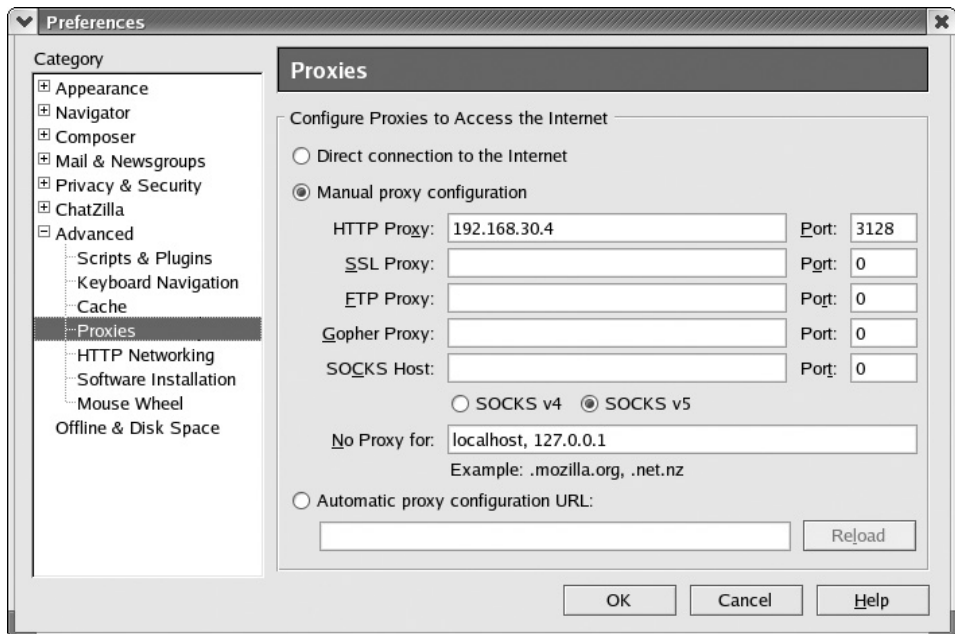
```
http_access allow lan_net
```

4. Configure access from your LAN to Squid with an appropriate **acl** command. The following command allows access from your **lan_net** with an IP network address of 172.168.30.0:

```
acl local_net src 172.168.30.0/255.255.255.0
```

5. Save your changes and exit.
6. Stop the Squid service if it isn't already running with the **service squid stop** command.
7. Create Squid swap directories with the **squid -z** command.
8. Start the Squid service with the **service squid start** command.
9. Configure a test client such as a Web browser to use your Squid service.

III 7-1



10. Test your client by using both HTTP and FTP addresses in the browser address. Use it to retrieve files from various sites on the Internet, such as www.redhat.com and ftp.kernel.org.

CERTIFICATION OBJECTIVE 7.03

The Very Secure FTP Service

FTP, the File Transfer Protocol, is one of the original network applications developed with the TCP/IP protocol suite. It follows the standard model for network services, as FTP requires a client and a server. The FTP client is installed by default on most operating systems, including Red Hat Enterprise Linux. If you've installed the FTP Server package group, you've installed the default Red Hat FTP Server, the very secure FTP (vsFTP) daemon. But before we begin, let's look at FTP clients.



Red Hat no longer includes alternate FTP servers such as WU-FTP. When you install vsFTP on your computer, you're installing the same FTP server used on ftp.redhat.com.

FTP Client (lftp)

The original FTP client software was a basic command line, text-oriented client application that offered a simple but efficient interface. Most Web browsers offer a graphical interface and can also be used as an FTP client.

Any FTP client allows you to view the directory tree and files. Using **ftp** as a client is easy. You could use the **ftp** command to connect to a server such as ftp.redhat.com with the following command:

```
# ftp ftp.redhat.com
```

The FTP client listed in the Red Hat Exam Prep guide is **lftp**. You can use it to connect to the FTP server of your choice. It automatically attempts an anonymous login. It also supports command completion, which can especially help you access files and directories with longer names.

Figure 7-7 illustrates a typical **lftp** session to ftp.redhat.com. As you can see, **lftp** uses a number of typical bash commands. The command completion feature lists available subdirectories. I've navigated to the directory with RHEL 3 Source RPMs.

Another advantage of **lftp** is that it can handle all of the basic upload and download commands of a regular FTP client. Some of these commands are as described in Table 7-1.

FIGURE 7-7

A typical lftp session

```
[root@Enterprise3 root]# lftp ftp.redhat.com
lftp ftp.redhat.com:~> ls
drwxr-xr-x  6 ftp      ftp      4096 Nov 11 16:18 pub
lftp ftp.redhat.com:> cd pub/
contrib/ redhat/ up2date/
lftp ftp.redhat.com:> cd pub/redhat/
ccm/  edk/  interchange/ mirror-tools/ support/
ccvs/ gnupro/ linux/      rhdb/      tux/
lftp ftp.redhat.com:> cd pub/redhat/linux/
1.0/ 2.1/ 4.1/ 5.1/ 6.1/ 7.1/ 8.0/ code/      preview/
1.1/ 3.0.3/ 4.2/ 5.2/ 6.2/ 7.2/ 9/  current@  rawhide/
2.0/ 4.0/ 5.0/ 6.0/ 7.0/ 7.3/ beta/  enterprise/ updates/
lftp ftp.redhat.com:> cd pub/redhat/linux/enterprise/3/en/os/i386/SRPMS/
cd ok, cwd=/pub/redhat/linux/enterprise/3/en/os/i386/SRPMS
lftp ftp.redhat.com:/pub/redhat/linux/enterprise/3/en/os/i386/SRPMS>
```

Almost all commands in FTP mode are run at the remote host, similar to a Telnet session. You can also run commands locally from the FTP prompt. When you start the command with an exclamation point (!), you can run regular shell commands.

This is only a subset of the commands available through **lftp**. Typing the **help** command will give you a full list of the available commands. The command **help cmd** yields a brief description of the command itself.

TABLE 7-1

Standard
FTP Client
Commands

Command	Description
cd	Changes the current working directory at the remote host
ls	Lists files at the remote host
get	Retrieves one file from the remote host
mget	Retrieves many files from the remote host with wildcards or full filenames
put	Uploads one file from your computer to the remote host
mput	Uploads a group of files to the remote host
pwd	Lists the current working directory on the remote host
quit	Ends the FTP session
!ls	Lists files on your host computer, in the current directory
lcd	Changes the local host directory for upload/download
!pwd	Lists the current working directory on local host computer

One graphical FTP client for Linux is GNOME FTP (GFTP). GNOME FTP provides an easy-to-use GUI interface to FTP. It also offers these features:

- Restartable transfers
- Multiple independent transfers
- Download file queuing
- Transferring whole directory trees (recursive transfers)
- Drag-and-drop transfer activation
- Session names and settings

And with GFTP you don't even need to know a single FTP command. If you have installed the `gftp` RPM, you can start the GFTP client from a GUI command line with the **gftp** command. Alternatively, you can start it from the Red Hat GNOME or KDE desktop with the Main Menu | Internet | More Internet Applications | gFTP command.

Installing the Very Secure FTP Server

The only FTP server included with RHEL 3 is vsFTP. In fact, it's the only RPM that's part of the FTP Server package group. If it isn't already installed, you probably won't have to bother with the GUI package management utility. This package has few dependencies, and should be installed automatically when you install RHEL 3. Assuming you've connected to the NFS installation share described in Chapter 2, you'd run the following command:

```
# rpm -Uvh /mnt/inst/RedHat/RPMS/vsftpd-1.2.0-4.i386.rpm
```

In the following sections, I'll show you how to start the vsFTP service the next time you reboot your computer, how it's already configured by default, and how to secure it by user. If you want to limit access to vsFTP by computer or network, you'll have to use the `tcp_wrappers` system, with its `hosts.allow` and `hosts.deny` files in the `/etc` directory, as described in Chapter 10.

Starting on Reboot

Once vsFTP is installed, you'll want to make sure it starts the next time you boot Linux. If it doesn't start when the person who grades your Red Hat exam reboots your computer, you may not get credit for your work configuring an FTP server.

The most straightforward way to make sure the FTP server starts the next time you boot Linux is with the **chkconfig** command. You'll need to set it to start in at least runlevels 3 and 5, with a command such as:

```
# chkconfig --level 35 vsftpd on
```

To see if the **chkconfig** command worked, use the **--list** switch:

```
# chkconfig --list vsftpd
```

If you want to start the vsFTP server, just start the service script with the following command:

```
# service vsftpd start
```

Basic FTP Server Configuration

As you've seen in Chapter 2, the vsFTP server is very easy to configure. Once you've started the daemon and modified any current firewalls, it's ready for use. By default, remote users can log in anonymously or with a username on the local computer. The sequence is straightforward. Assume you have a vsFTP server on a computer named Enterprise3. To connect anonymously using the **lftp** client, you'd run the following command:

```
# lftp Enterprise3
```

This opens a connection in the `/var/ftp` directory. It looks like a top-level root directory to the **lftp** client. Alternatively, if you wanted to connect to the account of user `michael`, you'd add just a little more:

```
# lftp -u michael Enterprise3
```

This opens a connection to the `/home/michael` directory. Unfortunately, the default is dangerous, as users who log in with a real username can navigate to the top-level root directory.

vsFTP Server Security

So now you know that vsFTP can be dangerous. But you can help secure your system by configuring vsFTP to disable logins from regular users. Naturally, you can configure vsFTP through the `vsftpd.conf` configuration file, in the `/etc/vsftpd` directory.

The commands in this file are straightforward. I urge you to read the file for yourself; the comments make many of the commands self-explanatory. Let's examine just a few of these commands in Table 7-2. I've focused on those commands that you might change to enhance the security of your system. The commands in the default vsftpd.conf file are just a small fraction of the commands that you can use. You can review the RHEL 3 Reference Guide available online at www.redhat.com/docs/manuals/enterprise for detailed information.

By default, vsFTP is configured to disable logins from sensitive users such as root, bin, and mail. The **userlist_enable=YES** command points to a list of disabled users in /etc/vsftpd.user_list. As vsFTP also uses Pluggable Authentication Modules (PAM) for security, it also disables the users in /etc/vsftpd.ftpusers. The list of users in the default versions of this file are identical.

If you want regular users to log into your FTP server, you should add the **chroot_local_user=YES** command. This helps secure your system by keeping regular users from navigating to your top-level root directory (/).

A couple of useful features for users help welcome them to your server. The following command provides a message for users who are logging into your system:

```
ftpd_banner=Welcome to blah FTP service
```

TABLE 7-2 Some vsFTP Server Configuration Commands

Command	Description
anonymous_enable=YES	If you don't want anonymous access, you'll have to set this to NO.
local_enable=YES	If you don't want regular users to log in, comment this out by adding a pound (#) character.
write_enable=YES	If you don't want remote users writing to your directories, comment out this command.
#chroot_list_enable=YES	If you set chroot_local_user=YES (see discussion that follows), you can configure users who are allowed to roam through your directories by activating this command.
pam_service_name=vsftpd	Configures Pluggable Authentication Module (PAM) security. For more information, see Chapter 10.
userlist_enable=YES	Don't change this! This keeps users such as root from logging into your system.
tcp_wrappers=yes	Supports the use of security commands in /etc/hosts.allow and /etc/hosts.deny. See Chapter 10.

This next message looks for a `.message` file in each directory, and sends it to the client:

```
dirmessage_enable=YES
```

Unfortunately, these messages don't work for users who log into your system using a client such as **lftp**.

Access by root and many service users are disabled by default. If you try to log in as root using **lftp**, it will look like you're connected. But when you try to do something as root, you'll see messages delaying your commands "before reconnect."

EXERCISE 7-5

Configuring a Basic FTP Server

In this exercise, you'll install and activate a basic vsFTP server on your RHEL 3 computer. While you should ideally also have a second computer, it isn't absolutely necessary here.

1. Check your system to see if the vsFTP server is installed. The easiest way is with the following command:

```
# rpm -q vsftpd
```
2. If it isn't already installed, use the techniques discussed in earlier chapters to install the vsFTP RPM package on your computer.
3. Activate the vsFTP server with the **service vsftpd start** command.
4. Make sure this server is automatically activated the next time you boot Linux with the following command:

```
# chkconfig --level 35 vsftpd on
```
5. Log into the vsFTP server as a regular user. You should preferably log in from a remote computer.
6. Once you're logged in, run the **cd ..** command twice (remember the space between the command and the two dots). Explore the local directory. You should see a danger here, as this is the root directory for the FTP server computer.
7. Close the FTP client session.

- 8. If you're concerned about the security issues, deactivate the vsFTP server. The best way to secure FTP is to uninstall the vsFTP server. However, if you need a vsFTP server, work with the lab at the end of the chapter.

CERTIFICATION OBJECTIVE 7.04

Mail Services



If you have Linux installed on your computer, you can set up a powerful mail server. Perhaps the most common mail server on the Internet is sendmail, which may already be installed on your RHEL 3 computer. Once it is installed and configured, you can set up sendmail as your own personal mail server (subject to the limitations of your ISP). One alternative to sendmail that is installed on RHEL 3 is known as postfix.

A number of alternatives to sendmail are not covered in this book; they include procmail, mail.local, exim, and uucp.

When you install sendmail and/or postfix, you also get huge and difficult-to-read configuration files. Do not be intimidated. There are only a few entries you are ever likely to have to change. A mail server has three major components, as described in Table 7-3. You need all three of these components to have a fully functional mail system.

On your Linux computer, you can configure your mail transfer agent (sendmail or postfix) for various outbound services, such as forwarding, relaying, method of transport (such as TCP or UDP), lists of computers with other MTAs, optional aliases, and spooling directories.

TABLE 7-3

Mail Server
Components

Abbreviation	Meaning	Examples
MTA	Mail transfer agent	sendmail, postfix, qmail
MUA	Mail user agent	mail, Mozilla, elm
MDA	Mail delivery agent	procmail, maildrop

E-mail systems are heavily dependent on name resolution. While you could handle name resolution through `/etc/hosts` on a small network, any mail system that requires Internet access needs access to a fully functional DNS server.

The `sendmail` and `postfix` systems use SMTP to send e-mail. But that is only one end of the mail system. You also need to configure POP and or IMAP to receive e-mail.

SMTP

SMTP, the Simple Mail Transfer Protocol, has become one of the most important service protocols of the modern era. Much of the Internet-connected world lives and dies by e-mail and relies on SMTP to deliver it. SMTP is a *protocol*, a set of rules for transferring data used by various Mail Transfer Agents. The default SMTP program for RHEL 3 is `sendmail`; the alternate is `postfix`.

Installing Mail Server Packages

The RPM packages associated with `sendmail` and `postfix` are both part of the Mail Server package group. It includes only eight RPM packages, as described in Table 7-4. You can install them with the `rpm` command or the Red Hat Package Management utility. You may not need to install all of these packages on your exam; read your requirements carefully!

Time is of the essence on the Red Hat exams. Since you may not need all of these packages, it may be faster to install these with the `rpm` command, especially if you're

TABLE 7-4 Mail Server Package Group RPMs

RPM Package	Description
sendmail	Installs the most popular mail server of the same name
imap	Supports both the IMAP and the POP incoming e-mail protocols
sendmail-cf	Adds a number of templates that you can use to generate your sendmail configuration file
spamassassin	Installs a program which you can start from sendmail or postfix to minimize spam
squirrelmail	Installs a Web-based e-mail server
postfix	Includes an alternative to sendmail
redhat-switch-mail redhat-switch-mail-gnome	Adds a GUI method for switching between sendmail and postfix

configuring your Linux computer from the text console. It takes time to start the GUI. On the other hand, once you've started the installation process, you can work on other exam requirements while the Package Management utility is installing your Mail Server packages. For more information on the Package Management utility, see Chapter 4.

If all that you're asked to configure from Table 7-4 during your Red Hat exam is sendmail, all that you would need to install is the sendmail and sendmail-cf RPM packages. And sendmail should already be installed by default, since it is part of the Base package group.



You can find a list of RPMs associated with each package group for an Intel 32-bit CPU installation in the `/usr/share/comps/i386/comps.xml` file.

sendmail

The sendmail daemon is configured from a directory of files in `/etc/mail` and a directory of configuration files in `/usr/share/sendmail-cf`. There are two basic configuration files: `sendmail.cf` for incoming mail and `submit.cf` for outgoing mail. I describe the key configuration files in `/etc/mail` in a bit of detail here.

- **sendmail.cf** The main sendmail configuration file.
- **sendmail.mc** A macro which you can edit and then generate a new `sendmail.cf` file.
- **access** Supports outgoing access control to your sendmail server. The default version of this file supports access from the local computer. You can add hostnames or networks to this list, with a message to REJECT with an error message, DISCARD without an error message, or RELAY to accept and send the e-mail.
- **domaintable** Allows you to map different domains. For example, if you've changed your domain name from Compaq.com to HP.com, people might still send e-mails to addresses such as michael@Compaq.com. The following line would forward that e-mail to michael@HP.com.

```
Compaq.com      HP.com
```
- **helpfile** Supports help commands when you manage your mail server from the sendmail prompt, which you can access with the **telnet localhost 25** command.
- **local-host-names** Allows you to add hostnames or aliases for your computer as a sendmail server. Enter one alias per line in this file.

exam**Watch**

*If you forget this detail,
run the `rpm -qi spamassassin` command.
You'll see it in the description.*

- **mailtable** Rarely used.
- **Makefile** Supports compiling the `sendmail.mc` file.
- **spamassassin** A directory which includes configuration files that can help you minimize spam. If you want to configure it on locally received e-mail, add the following line to `/etc/procmailrc`:

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-default.rc
```

- **statistic** Collects statistics on sendmail usage in binary format. You can read it with the **mailstats** command.
- **submit.cf** The main outgoing sendmail configuration file.
- **submit.mc** A macro which you can edit and then generate a new `submit.cf` file.
- **trusted-users** Lists special users which can send e-mail without warnings. For example, you saw e-mail addresses in the Apache configuration file; if you include the `apache` user in this list, it can send messages to your Web server administrators without generating sendmail warning messages.
- **virtusertable** Supports e-mail forwarding; if you have users outside your network who use your sendmail server, you can enter individual e-mail addresses or domains for allowed users.

If you don't see some of these files or directories, you may not have both sendmail RPM packages installed: `sendmail` and `sendmail-cf`. Use the `rpm -q packagename` command to check if you have these packages installed, and install them as required.

Configuring sendmail for Basic Operation

When sendmail starts, it reads the `/etc/sendmail.cf` and `/etc/submit.cf` files. The `sendmail.cf` file is a long (around 1,800 lines) file that may seem difficult to decipher but includes a wealth of helpful comments. The `submit.cf` file is nearly as long. This file provides detailed rules (organized into rulesets) on how sendmail should process e-mail addresses, filter spam, talk to other mail servers, and more.

This file is extremely complex and uses cryptic syntax. Fortunately, most of the directives included in this file are standards that you don't need to change. Many are

required by various Internet agreements relating to e-mail address, mail transfer agents, and more.

Red Hat tries to simplify this process with a smaller file, `/etc/mail/sendmail.mc`, which contains only the most relevant configuration directives. It is composed entirely of macros that define key `sendmail.cf` settings. Once you've configured this file, you can use the **make** command to compile a new, custom `sendmail.cf` file. However, the default RHEL 3 version of this file is still over 140 lines long. Remember, the Red Hat Exam Prep guide only requires that you configure the service for basic operation. Therefore, I'll highlight those commands that you may want to change or modify for that purpose.

While older versions of the `submit.mc` file required work, the default versions included with RHEL 3 (or Red Hat Linux 9) should work without changes. Unless you're on an IPv6 network, you can focus entirely on the `sendmail.mc` file. If you are on an IPv6 network, you'll need to change the loopback address in the last line as described in a comment in the file.

The `sendmail.mc` file is made up of directives (macros) used to create content for `sendmail.cf`. (Other related `.mc` files follow the same criteria and use many of the same commands). These macros do the following:

- Add comments to aid in comprehension
- Define key variables and values
- Enable or disable features
- Create variables with specific settings

The most basic macro is **dnl**, which tells **m4** to delete from this point through to the end of the line. It is used to comment out descriptive text or disable a feature that would otherwise be included.

The **include** directive instructs the **make** command to read the contents of the named file and insert it at the current location in the output. This is how additional configuration information (needed by `sendmail` but not relevant to mail configuration) is kept separately from settings you may wish to change.

The **define** directive sets files or enables features that you wish to use. In the preceding example, you set the path to your e-mail name user `ALIAS_FILE` as `(/etc/aliases)`, identify where `procmail` lives (`PROCMAIL_MAILER_PATH`), and provide the path for the official database of e-mail users—in this case, in `virtualusertable.db`.

The **FEATURE** directive enables specific features. For example, you use **FEATURE** to set `accept_unresolvable_domains`. This allows you to accept mail where you can't figure out the domain of the user who sent the e-mail. Specifically, an unresolvable

domain refers to a case where it isn't possible to find a domain name through a reverse IP address lookup. If you don't have reliable DNS access, you may need this feature, or else your sendmail configuration may refuse a lot of valid e-mail.

DAEMON_OPTIONS directly controls the SMTP daemon. In the default case, you do not accept any mail from outside your own system (note the loopback address on this line).

You'll notice unusual quote characters in most of these lines. The command inside parentheses starts with a back quote (`) and ends with a single quote (').

Configuring and Securing sendmail

In this section, you'll modify the `sendmail.mc` configuration file. Back it up first! You only need to make a couple of adjustments to get your system ready for use on the Internet. By default, the following line limits sendmail use to the local computer:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

You can allow other computers to use your sendmail server by commenting out this line. As described earlier, this requires a **dnl** command. The result would be:

```
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

Next, if you have reliable DNS access, comment out the **FEATURE** directive that allows you to **accept_unresolvable_domains**. This blocks spammers who use just an IP address or spammers who fake their domain name to hide themselves.

```
FEATURE(`accept_unresolvable_domains')dnl
```

But that's not enough. If you want to allow remote computers or networks access to your sendmail server, you'll need to add their names or addresses to the `/etc/mail/access` file. For example, if you wanted to allow access to the 192.168.30.0 domain, you'd add the following line to that file:

```
192.168.30 RELAY
```

Watch the notation; unlike other services, there is no dot (.) at the end of the address. It covers all computers on the 192.168.30.0 network. Alternatively, you could designate the `example.com` domain, or a specific computer name or IP address.

Back up the current `sendmail.cf` file. Then you can generate a new `sendmail.cf` file and restart sendmail services with the following command:

```
# cp /etc/mail/sendmail.cf /root
# make -C /etc/mail/
```



Now you can reconfigure your e-mail clients such as Mozilla or even Microsoft Outlook Express to send outgoing e-mail through your sendmail server. You'll need to set your sendmail computer domain name or IP address as the SMTP outgoing mail server.

In previous versions of sendmail, all you needed to do was process `sendmail.mc`; it was therefore sufficient to use the `m4` macro command. But you may be processing more files. The `make -C /etc/mail` command shown processes all files in the `/etc/mail` directory.

Configuring sendmail to Start at Reboot

Now start or restart sendmail to make sure it reads your new `sendmail.cf` configuration file:

```
# chkconfig sendmail --level 35 on
# service sendmail restart
```

Your sendmail (SMTP) service should now be up and running and ready to accept mail from any (valid) source.

Troubleshooting sendmail

When name resolution is not working on your network, sendmail doesn't know where to send your outbound e-mail. These messages are placed in a queue that tries to resend your e-mail at regular intervals. Other mail forwarders and relay hosts on the Internet provide the same functionality if a network segment is not working. As an administrator, you need to monitor this queue. If it gets overloaded, you may wish to reconfigure messages for that network to be sent at more irregular times. See the following code for an example of a problem message:

```
# mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/root" 1 messages 1 new
>N 1 MAILER-DAEMON@localh Wed Dec 3 08:55 60/1914 "Returned mail: see tr"
&    # simply press <ENTER> key to see each message
Message 1:
From MAILER-DAEMON@localhost.localdomain Wed Dec 3 08:55:39 2002
Date: Wed, 3 Dec 2003 08:55:39 -0500
From: Mail Delivery Subsystem <MAILER-DAEMON@localhost.localdomain>
To: root@localhost.localdomain
MIME-Version: 1.0
Content-Type: multipart/report; report-type=delivery-status;
        boundary="DAA03153.938948139/localhost.localdomain"
```

42 Chapter 7: Linux Sharing Services

```
Subject: Returned mail: see transcript for details
Auto-Submitted: auto-generated (failure)
This is a MIME-encapsulated message
--DAA03153.938948139/localhost.localdomain
The original message was received at Wed, 3 Dec 2003 08:55:39 -0500
from Enterprise3
--- The following addresses had transient non-fatal errors ---
<michael@mommabears.cob>
    (reason: 550 Host unknown)
    --- Transcript of session follows ---
550 5.1.2 <michael@mommabears.cob>... Host unknown
(Name server: mommabears.cob: host not found)
..
(additional details deleted)
..

& d    # delete current read buffer
& q    # quit mail
#
```

In the previous example, the destination name server (mommabears.cob) could not be resolved (it is mommabears.com). Consequently, sendmail notifies the sender (root@localhost.localdomain) that the mail could not be delivered.

In this case, the problem is straightforward: you've specified a nonstandard domain. However, you may get a similar error for one of the following reasons:

- No DNS server available, if so required in sendmail.mc.
- Access from a computer not listed or specifically denied in the /etc/mail/access configuration file.
- Some firewall blocking access. You can configure a firewall to allow incoming e-mail through TCP/IP port 25. You can find more information about this in Chapter 10.

Configuring and Activating postfix

You can configure the postfix mail server as a substitute for sendmail. The configuration files are stored in the /etc/postfix directory. The main configuration file, main.cf, is somewhat simpler than sendmail.cf, as it includes around 600 lines. Back up this file

and open it in a text editor. There are several things that you need to configure in this file to get it working:

- Activate and modify the following **myhostname** command to point to the name of your computer:

```
#myhostname = host.domain.tld
```

- Activate and modify the following **mydomain** command to reflect the domain name or IP network of your system (if you substitute your IP network, use the same format as illustrated for the sendmail.mc file):

```
#mydomain = domain.tld
```

- The following two commands set postfix to listen to only the local computer. Activate and deactivate the following commands to remove that limitation:

```
#inet_interfaces = all
inet_interfaces = localhost
```

- Activate and modify the following **mynetworks** command to point to the IP network address that you want to serve.

```
#mynetworks = 168.100.189.0/28, 127.0.0.0/8
```

Configuring postfix to Start at Reboot

Now start or restart postfix to make sure it reads your new /etc/postfix/main.cf configuration file:

```
# chkconfig postfix --level 35 on
# service postfix restart
```

Your postfix service should now be up and running and ready to accept mail from any (valid) source.

Mail Clients

One of the prerequisite skills for the Red Hat exams is to configure an e-mail client. This should be easy for anyone who has used Linux for any length of time. In fact, the process for graphical Linux clients such as Evolution or Mozilla is quite similar to the process for various Microsoft e-mail clients. But as you've seen earlier in this section, there are text-based mail clients as well.

Command Line Mail

To test your mail system, you can use the built-in command line **mail** utility, a simple text-based interface. The system keeps each user's mail in a system directory. Once users read a message, they can reply, forward, or delete it. If they do not delete the message before quitting the **mail** utility, the system stores the message in the `/var/mail` directory, in a file named after the applicable username.

You can certainly use any of the other mail readers, such as **mutt**, or the e-mail managers associated with different GUI Web browsers to test your system. Other mail readers store messages in different directories. For example, **pine** would create and store messages for user **mj** in the `/home/mj/mail` directory.

To send mail to another user, you can use the **mail** command line utility. There are two basic methods for using **mail**. First, you can enter the subject and then the text of your message. When you're done, press CTRL-D and then enter another addressee in the Cc: line, if desired. When you press ENTER, the message is sent and the **mail** utility stops and sends you back to the command line.

```
$ mail Michael
Subject: Test Message
Sent and received
Cc: mjang@example.com
$
```

Alternatively, you can redirect a file as the text of an e-mail to another user. For example, the following command sends a copy of `/etc/hosts` to the root user, with the Subject name of "hosts file":

```
$ mail root@localhost -s 'hosts file' < /etc/hosts
```

Reading Mail Messages

By default, the mail system doesn't open unless you actually have e-mail in your in box. Once it is open, you'll see a list of new and already read messages. To read a specific message, enter the number of the message and press ENTER. If you press ENTER with no argument, the mail utility assumes you want to read the next unread message. To delete a mail message, use the **d** command after reading the message, or use **d#** to delete the message numbered #.

Mail Group "Alias" Lists

If you have a distribution list of people for the same e-mail, you can set it up in the `/etc/aliases` file. By default, it's set up to forward e-mail from pseudo-accounts such as **system** and **apache** to **root**. You can change it by adding a group list similar to the following:

```
groupname: user01, user02, othergroupname
```

You can then run the **newaliases** command to compile this database. Then all you need to do is name the group of users as addressees for your e-mail.

POP

The sendmail and postfix services provide their own incoming e-mail support. If you've started them on your server, you won't need to activate the ipop3d service. You can enable the ipop3d service by editing the `/etc/xinetd.d/ipop3` configuration file.

The Post Office Protocol (POP) is one of the two major mail delivery protocols. It includes some basic commands that allow you or an e-mail client to send and retrieve messages. A mail service can be configured to be a central depository for incoming mail messages from any other MTA service. Client applications then download the mail messages off the POP server for processing at the local host. The ipop3d service handles all requests.



You can configure user accounts that are only designed to service POP user accounts, where users log in and receive mail only, and no interactive service is provided. Just set up the appropriate mail client in the login configuration sequence for a given user.

IMAP

The IMAP service is the other major mail delivery protocol. While POP downloads all e-mail to the client, an IMAP server maintains all mail messages on the server, as a database. IMAP is commonly used by businesses that service users who log in from different locations. It's also the most common mail delivery protocol for Web-based mail services.

Installing, Configuring, and Securing POP and IMAP

Both POP3 and IMAP4 servers are part of one Red Hat RPM package: `imap-2002d-2.i386.rpm`. Install it as required. No special configuration is needed. It includes SSL support; it's also one of the xinetd "super servers." These packages are configured in the `/etc/xinetd.d` directory. POP3 service is configured through the `ipop3` file; IMAP4 service is configured through the `imap` and `imaps` files. All of these are disabled by default with the following line:

```
disable = yes
```

You can enable it in one of two ways: edit the file directly to change this line to **disable = no**. Alternatively, the following command activates the POP3 service:

```
# chkconfig ipop3 on
```

CERTIFICATION SUMMARY

You can configure a number of network sharing services on your RHEL 3 computer. Apache is the most important Web server on the Internet. Squid allows you to save bandwidth. The very secure FTP daemon allows you to let users transfer files to and from your server efficiently and securely. Mail services enable you to set up e-mail send and receive functions on your network.

Apache was developed from the NCSA Web server. Once the appropriate packages are installed, you have a structure and sample Web pages in the `/var/www/html` directory, based on the `/etc/httpd/conf/httpd.conf` configuration file. The `httpd.conf` file is organized in containers. You can create virtual hosts for multiple Web sites on your computer, even if you have only one IP address.

Squid is a proxy server that allows a network to filter its HTTP and FTP traffic through a cache. Requests are taken from the cache when possible. This reduces the load between the LAN and the Internet, reducing your network costs. When users access cached files, they get better performance from the external network.

FTP has two parts: the client and the server. FTP clients such as **ftp** and **lftp** are available from the command line. Red Hat includes one FTP server, the very secure FTP service. You can configure it in detail through the `/etc/vsftpd/vsftpd.conf` configuration file.

Red Hat includes two mail servers with RHEL 3: **sendmail** and **postfix**. Both rely on SMTP to send e-mail over the Internet. The **sendmail** service includes difficult-to-read configuration files: `sendmail.cf` and `submit.cf`, both in the `/etc/mail` directory. Fortunately, you can configure these files through easier-to-read macro files, `sendmail.mc` and `submit.mc`. **postfix** is somewhat easier to configure directly through its own configuration file, `/etc/postfix/main.cf`. Red Hat also provides a variety of e-mail clients, including **mail**, **mutt**, and the mail clients that come with the various Web browsers that are also available for Linux. These clients either use POP or IMAP protocols to receive e-mail.

Because RHCE is a performance-based exam, it is important to practice all the skills discussed in this chapter. You may need to use these skills on the exam!



TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 7.

Apache

- ☐ Red Hat Enterprise Linux 3 includes the Apache Web server, which is currently used by over twice as many Internet Web sites as all other Web servers combined.
- ☐ Web services are an easy way to provide simple, secure access to many types of documents. The Apache Web server provides both normal and secure Web services using the HTTP and HTTPS protocols.
- ☐ With Apache 2.0, you can configure multiple Web sites on your server, even if you have only one IP address. This is possible through the use of virtual hosts.

Easy Squid Configuration

- ☐ Squid is a high-performance HTTP and FTP caching proxy server.
- ☐ The main Squid configuration file is long, but all you need to do in `/etc/squid/squid.conf` is configure the following parameters: **visible_hostname**, **http_access**, and **acl**.
- ☐ Squid can refer requests to sibling and parent proxy servers. If the request still isn't available, a parent proxy server refers the request to the Internet.
- ☐ Once Squid is configured, you can set each computer on the LAN to browse Web pages to the proxy server on port 3128.

The Very Secure FTP Service

- ☐ The FTP, or File Transfer Protocol, service includes a client and a server. RHEL 3 includes the command line **ftp** and **lftp** clients.
- ☐ RHEL 3 includes the vsFTPd server. The default configuration allows anonymous and real user access.
- ☐ You can customize vsFTPd through the `/etc/vsftpd/vsftpd.conf` configuration file. It also uses authentication files in the `/etc` directory: `vsftpd.ftpusers`, `vsftpd.user_list`, and `vsftpd.chroot_list`.

Mail Services

- ☐ RHEL 3 includes sendmail and postfix, two mail servers that you can use to support sending and receiving e-mail.
- ☐ The main sendmail configuration file is `/etc/mail/sendmail.cf`. It's easier to configure sendmail through its macro file, `/etc/mail/sendmail.mc`.
- ☐ You can customize the computers allowed to access your sendmail server through the `access` and `virtusertable` files in the `/etc/mail` directory.
- ☐ The postfix server is easier to configure through configuration files in the `/etc/postfix` directory. In fact, you can configure the `main.cf` file directly.
- ☐ Different mail clients can receive e-mail using the POP or IMAP protocols. Mail clients such as **mail** and **mutt** are available at the command line. GUI mail clients are also available, usually as part of a Web browser package.

SELF TEST

The following questions will help you measure your understanding of the material presented in this chapter. Read all the choices carefully, as there may be more than one correct answer. Don't focus exclusively on these questions. There are no longer any multiple choice questions on the Red Hat exams. These questions exclusively test your understanding of the chapter. While the topics in this chapter are "prerequisites," it is okay if you have another way of performing a task. Getting results, not memorizing trivia, is what counts on the Red Hat exams.

Apache

1. Which of the following Apache directives give you the base directory for configuration and log files?
 - A. ServerRoot
 - B. DocumentRoot
 - C. Options
 - D. UserDir
2. How would you configure a virtual host as a secure server?
 - A. Set up password protection on the DocumentRoot.
 - B. Configure security in `/etc/httpd/conf/httpd.conf`.
 - C. Create a Virtual Host container in `/etc/httpd/conf.d/ssl.conf`.
 - D. Add the contents of the `ssl.conf` file to the `httpd.conf` configuration file.
3. Once you've modified `httpd.conf`, which of the following commands should you use to reread this file?
 - A. `service httpd restart`
 - B. `service apache reload`
 - C. `apachectl reload`
 - D. `apachectl restart`
4. Which of the following commands creates a password file and configures a password for user `elizabeth`?
 - A. `.htpasswd elizabeth`
 - B. `htpasswd -c /etc/httpd/passwords elizabeth`
 - C. `passwd elizabeth`
 - D. `service htpasswd elizabeth /etc/httpd/passwords`

Easy Squid Configuration

5. Which of the following commands in the squid.conf file is used to point to the network IP address served by Squid?
 - A. `visible_hostname`
 - B. `http_access allow`
 - C. `acl`
 - D. `squid -z`
6. Which of the following commands should you use before starting Squid for the first time? This command configures the Squid cache directories.
 - A. `visible_hostname`
 - B. `http_access allow`
 - C. `acl`
 - D. `squid -z`

The Very Secure FTP Service

7. Which of the following commands should you disable if you don't want users logging into their accounts through the vsFTP server?
 - A. `local_enable = yes`
 - B. `anonymous_enable = yes`
 - C. `nopriv_user = ftpsecure`
 - D. `chroot_local_user = yes`
8. Which of the following commands should you enable if you want to keep regular users from getting to the top-level root directory (/) on your computer?
 - A. `local_enable = yes`
 - B. `anonymous_enable = yes`
 - C. `nopriv_user = ftpsecure`
 - D. `chroot_local_user = yes`

Mail Services

9. Which of the following directives would you disable in /etc/mail/sendmail.mc to allow your sendmail server to serve more than just the local computer?
 - A. `FEATURE(`mailertable',`hash -o /etc/mail/mailertable.db')dnl`
 - B. `FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable.db')dnl`

- C. LOCAL_DOMAIN(`localhost.localdomain`)dn1
 - D. DAEMON_OPTIONS(`port=smtp,Addr=127.0.0.1, Name=MTA`)dn1
- 10.** Which of the following commands should you add to `/etc/mail/access` if you want to set up service to the 10.11.12.0 network?
- A. 10.11.12.0/24 RELAY
 - B. 10.11.12 RELAY
 - C. 10.11.12. RELAY
 - D. 10.11.12.0/255.255.255.0 RELAY

LAB QUESTIONS

Lab 1

In this first lab, you'll install and configure Apache to start and run automatically the next time you boot your computer. You'll also configure the default Mozilla Web browser home page as the default home page for the local computer.

Lab 2

In this second lab, you'll configure two Web sites on the local Apache server. Call them `big.example.big` and `small.example.small`. Don't forget to create the directories that you need, as well as set up these Web sites on your DNS server or `/etc/hosts` file. Make sure your Web sites are accessible to users from remote computers on your network. Add an appropriate `index.html` file to the `DocumentRoot` for each Web site. Simple Web pages, such as a single line of text, are acceptable.

Lab 3

Continuing on with Apache, now configure secure versions for each of your two Web sites. Make sure that there are appropriate directories available for each secure Web site.

Lab 4

Set up a Squid proxy server on your computer. Set up access to your LAN on the 10.11.12.0/255.255.255.0 network. Assign appropriate values to `acl`, `http_access`, and `visible_hostname`. Set up the cache directories for Squid. Make sure it starts now, and automatically the next time you reboot your computer.

Lab 5

Configure an FTP server for your computer. Make sure to allow only anonymous access. Don't allow anonymous users to upload to your server. Enable messages when users access your `/var/ftp` and `/var/ftp/pub` directories. Add an appropriate one-line message to each directory. Test the result, preferably from a remote computer. Make sure to start the vsFTP server now, and see that it starts automatically the next time you reboot your computer.

Lab 6

Set up a sendmail mail server for your network. First, make sure to disable local-only access in the `/etc/mail/sendmail.mc` file. Add your network to the `/etc/mail/access` file. Test the result, preferably from a remote computer on your network. Configure Mozilla to read your e-mail. You can set it up to read e-mail without downloading it from the server (even if it's a POP3 server). Make sure to start the sendmail server now, and see that it starts automatically the next time you reboot your computer.

SELF TEST ANSWERS

Apache

1. ☒ **A** is correct. The **ServerRoot** directive sets the default directory for the Apache server. Any files and directories not otherwise configured—or configured as a relative directory—are set relative to **ServerRoot**.
☒ **B**, **C**, and **D** are incorrect. The **DocumentRoot** is the root directory for each Web site. The **Options** directive is used to set exceptions to document permissions. The **UserDir** directive supports access to Web sites from users' home directories.
2. ☒ **C** is correct. The Red Hat Apache configuration supports the creation of secure servers in `/etc/httpd/conf.d/ssl.conf`. You can configure virtual hosts for your secure Web site in the same way as you can in `/etc/httpd/conf/httpd.conf`.
☒ **A**, **B**, and **D** are incorrect. Password protection on a Web site is unrelated to a secure Web site using the HTTPS protocol. While you can configure a secure Web site in `httpd.conf`, the configuration file designed for this purpose is `ssl.conf`. For the same reason, you should not add the contents of `ssl.conf` to `httpd.conf`.
3. ☒ **D** is correct. The **apachectl** command is the preferred method to control the Apache service.
☒ **A**, **B**, and **C** are incorrect. While the **service** command is generally used to control Linux services, it does not control Apache gracefully. The name of the Apache service is `httpd`. And there is no **apachectl reload** command.
4. ☒ **B** is correct. The **htpasswd** command sets up passwords. The **-c** switch is required to create a new file to store passwords.
☒ **A**, **C**, and **D** are incorrect. The `.htpasswd` file is not a command. The **passwd** command isn't used for Web server usernames or passwords. The **service** command has nothing to do with passwords.

Easy Squid Configuration

5. ☒ **C** is correct. The **acl** command in `squid.conf` can be used to specify the network to be served by Squid.
☒ **A**, **B**, and **D** are incorrect. The **visible_hostname** command is used to identify the local computer. The **http_access** command can be used to actually configure Squid for the network of your choice. The **squid -z** command sets up the directories for the Squid cache.
6. ☒ **D**. See answer 5 for the reasons.
☒ **A**, **B**, and **C** are incorrect. See answer 5 for the explanation.

The Very Secure FTP Service

7. ☒ A is correct. The `local_enable` command allows local users to log in remotely to their home directories. Disabling this command disables this functionality.
☒ B, C, and D are incorrect. The `anonymous_enable = yes` command supports access by the anonymous user. The `nopriv_user = ftpsecure` command is already disabled in the default version of `vsftpd.conf`; it defines an unprivileged user. The `chroot_local_user = yes` command does not keep users with local accounts from logging in, it just keeps them from accessing the root directory of your computer.
8. ☒ D is correct. See answer 7 for the reasons.
☒ A, B, and C are incorrect. See answer 7 for the explanation.

Mail Services

9. ☒ D is correct. The `DAEMON_OPTIONS` directive in the default `sendmail.mc` file prevents access to anything but the local computer.
☒ A, B, and C are incorrect. Answers A and B refer to database files which allow you to configure access to other computers, therefore you should not disable these commands. Answer C just identifies the domain name of the local computer.
10. ☒ B is correct. This is the required format in the `/etc/mail/access` configuration file.
☒ A, C, and D are incorrect. These options do not conform to the required format in `/etc/mail/access`.

LAB ANSWERS

Lab 1

First, make sure the Apache Web server is installed. If an `rpm -q httpd` command tells you that it is missing, you haven't installed the Web Server package group. The most efficient way to do so is with the Red Hat Package Management utility, only accessible through the Linux GUI.

To configure Apache to start, you'll want to run the `apachectl start` command. To make sure it starts the next time you boot your computer, you'll want to run the `chkconfig --level 35 httpd on` command.

Once Apache is installed, you should be able to access it by opening a browser and navigating to `http://localhost`. You can see in the default Apache configuration file that the DocumentRoot is located in `/var/www/html`. The default Mozilla home page is located at `/usr/share/doc/HTML/index.html`. You can copy that `index.html` file to the `/var/www/html` directory, and test the result by navigating once

again to `http://localhost`. If you did not copy the other files associated with the default Mozilla home page, you'll be missing some icons.

Lab 2

This lab requires that you create two virtual hosts in the main Apache configuration file, `/etc/httpd/conf/httpd.conf`. To do so, you should do the following:

1. Set the **NameVirtualHost** directive to the IP address and port (80) serving your intended network audience.
2. Add a **VirtualHost** container with the same IP address.
3. Assign the **ServerAdmin** to the e-mail address of this Web site's administrator.
4. Configure a unique **DocumentRoot** directory.
5. Set the first **ServerName** to `big.example.big`.
6. Add **ErrorLog** and **CustomLog** directives, and set them to unique filenames in the `/etc/httpd/logs` directory. With the default **ServerRoot**, you can use a relative logs directory, such as:

```
ErrorLog logs/big.example.big-error_log
```

7. Make sure to close the **VirtualHost** container.
8. Repeat the process for the second Web site, making sure to set the second **ServerName** to `small.example.small`.
9. Close and save the `httpd.conf` file with your changes.
10. Create any new directories that you configured with the **DocumentRoot** directives.
11. Create `index.html` text files in each directory defined by your new **DocumentRoot** directives. Don't worry about HTML code; a text file is fine for the purpose of this lab.
12. Make sure these domain names are configured in your DNS server or in `/etc/hosts`. For example, you could add the following lines to `/etc/hosts`:

```
192.168.30.2 big.example.big
192.168.30.2 small.example.small
```

13. Disable any firewall on your computer. Alternatively, you can use the Security Level Configuration tool (**redhat-config-securitylevel**) utility to allow HTTP data through your firewall; see Chapter 10 for more information on this process.

14. Finally, make sure to run the **apachectl restart** command to reread the `httpd.conf` configuration file, so Apache reads your changes.
15. Now you can test the result in the browser of your choice. If it works, the `big.example.big` and `small.example.small` domain names should direct you to the `index.html` files that you created for each Web site.

Lab 3

The basics of this lab are straightforward. You'll need to repeat the same basic steps as you performed in Part 2; you're just editing the `/etc/httpd/conf.d/ssl.conf` configuration file. However, there are a few things to be concerned about:

- Make sure that the top **VirtualHost** directive points to the IP address that you're using for your Web server.
- Set up the `DocumentRoot` in a directory different from a regular Web server.
- Configure the `ErrorLog` and `CustomLog` separately; it can help to associate it with the name of the secure Web site.

Continuing on with Apache, now configure secure versions for each of your two Web sites. Make sure that there are appropriate directories available for each secure Web site.

Lab 4

First, Squid is automatically installed when you install the Web server package group. To configure a Squid proxy server for your network, you'll need to configure `/etc/squid/squid.conf`. Assume the name of your computer is `myproxy`, and you're arbitrarily assigning `mylan` as the name for your LAN. If your network IP address is not `10.11.12.0`, substitute accordingly. In this file, you'll need to add the following lines:

```
visible_hostname=myproxy
acl mylan src 10.11.12.0/255.255.255.0
http_access allow mylan
```

Next, you'll need to set up the Squid directories with the following command:

```
# squid -z
```

Finally, to configure Squid to start, you'll want to run the **service squid start** command. To make sure it starts the next time you boot your computer, you'll want to run the **chkconfig --level 35 squid on** command.

But you'll also need to activate proxy server access in client applications such as Web browsers. Remember that you can do so by pointing your browsers to port 3128.

Lab 5

The vsFTP server is part of a one-RPM package group. So if you have not installed this server during the installation process, the quickest thing to do is to connect to your installation source (CD or network), and install it from that location. For example, if the source is mounted on /mnt/source, you'd install it with the following command:

```
# rpm --Uvh /mnt/source/RedHat/RPMS/vsftpd-1.2.0-4.i386.rpm
```

This also installs configuration files in the /etc and /etc/vsftpd directories. The main configuration file is /etc/vsftpd/vsftpd.conf. Based on the RHEL 3 default version of this file, you can make the following changes. To allow only anonymous access, comment out the following line:

```
local_enable=yes
```

Anonymous users are already prevented from uploading files to your server. You could enable it by activating the **anon_upload_enable=yes** command. By default, messages are already enabled for directory access on an FTP server, courtesy of the following command:

```
dirmessage_enable=yes
```

Actually configuring a message is a matter of creating a text file, and saving it as .message in the desired directories, /var/ftp, and /var/ftp/pub. You could add a simple line such as “root directory for the FTP server” or “main download directory.”

Finally, to configure the Red Hat FTP server to start, you'll want to run the **service vsftpd start** command. To make sure it starts the next time you boot your computer, you'll want to run the **chkconfig --level 35 vsftpd on** command.

Lab 6

The sendmail mail server is part of the Mail Server package group. It is automatically installed when you install RHEL 3. You could install related packages using the Package Management utility, or you could install the sendmail-cf RPM package.

To disable local-only access in the /etc/mail/sendmail.mc file, comment out the following line. Unlike most Linux configuration files, the comment code is a **dnl** at the start of this line:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA') dnl
```

The **dnl** at the end of the line does not affect the command to its left. Next, you'll want to enable support through /etc/mail/access. If you want to support your LAN with this server, and its network address is 10.11.12.0, you'd add the following command line to /etc/mail/access:

```
10.11.12 RELAY
```

You can test the result from the e-mail client of your choice. It's easiest to use a client such as Mozilla. Netscape users should find this browser to be familiar; it uses the Netscape code, which has been released under an open source-style license. If you're not familiar with Mozilla, you can open it by clicking on the globe icon adjacent to the Main Menu icon in the lower-left corner of the desktop. Once Mozilla is open, click Window | Mail & Newsgroups to open the Mozilla mail management utility. If this is the first time you've opened the Mozilla mail management utility, the Account Wizard prompts you to add the information associated with your e-mail account. Otherwise, click the Add Account button.

You'll need to know the name of your incoming mail server, whether it conforms to the POP3 or IMAP4 protocols, and the name (and password) of your account. You presumably already know the name of the outgoing mail server, the name of the computer with the sendmail server that you just configured.